

UNCLASSIFIED

AD NUMBER

ADA800039

LIMITATION CHANGES

TO:

Approved for public release; distribution is unlimited.

FROM:

Distribution authorized to DoD only;
Administrative/Operational Use; 01 FEB 1952.
Other requests shall be referred to Office of
Naval Research, 875 North Randolph Street,
Arlington, VA 22203. Pre-dates formal DoD
distribution statements. Treat as DoD only.

AUTHORITY

ONR ltr dtd 27 Apr 1966

THIS PAGE IS UNCLASSIFIED

UNCLASSIFIED

AD NUMBER	
ADA800039	
CLASSIFICATION CHANGES	
TO:	unclassified
FROM:	restricted
LIMITATION CHANGES	
TO: Distribution authorized to DoD only; Administrative/Operational Use; 01 FEB 1952. Other requests shall be referred to Office of Naval Research, 875 North Randolph Street, Arlington, VA 22203. Pre-dates formal DoD distribution statements. Treat as DoD only.	
FROM: Controlling DoD Organization: Office of Naval Research, 875 North Randolph Street, Arlington, VA 22203.	
AUTHORITY	
E.O. 10501 dtd 5 Nov 1953; Pre-dates formal DoD distribution statements. Treat as DoD only.	

THIS PAGE IS UNCLASSIFIED

REEL-C 7402

A.T.I 209023

Reproduced by

D O C U M E N T S E R V I C E C E N T E R
ARMED SERVICES TECHNICAL INFORMATION AGENCY
KNOTT BUILDING, DAYTON, 2, OHIO

"NOTICE: When Government or other drawings, specifications or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the U.S. Government thereby incurs no responsibility, nor any obligation whatsoever; and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications or other data is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use or sell any patented invention that may in any way be related thereto."

UNCLASSIFIED

ATI No. **209023**
ASTIA FILE COPY

with
ber 1953

Service Center
ices Tech. Info Agency

RESTRICTED

209023-1

**FEASIBILITY OF ACTUATING
TRAINERS BY DIGITAL
COMPUTERS**

RETURN TO:
ASTIA REFERENCE CENTER
LIBRARY OF CONGRESS
WASHINGTON 25, D.C.



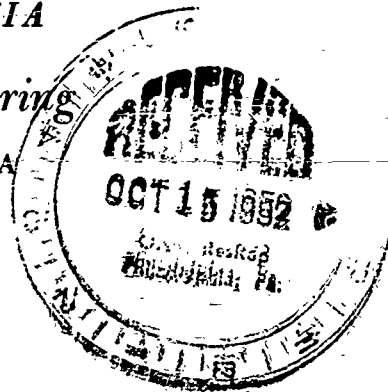
LIBRARY OF CONGRESS
REFERENCE DEPARTMENT
TECHNICAL INFORMATION DIVISION
FORMERLY
(NAVY RESEARCH SECTION)

OCT 20 1952

UNIVERSITY of PENNSYLVANIA
Moore School of Electrical Engineering
PHILADELPHIA, PENNSYLVANIA

1 February 1952

Research Division Report 52-22



RESTRICTED

Copy 2

RESTRICTED

Progress Report No. 1

FEASIBILITY OF ACTUATING TRAINERS
BY DIGITAL COMPUTERS

Report of Work under Contract N6onr-24915

Project 24-F-2

between

Office of Naval Research
Special Devices Center
Port Washington, New York

and

University of Pennsylvania
Moore School of Electrical Engineering
Philadelphia 4, Pennsylvania

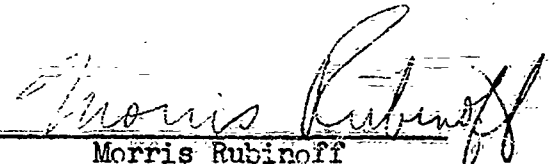
1 February 1952
Research Division Report 52-22

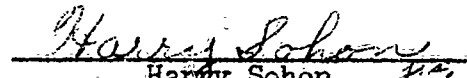
RESTRICTED

The following members of the Research Division, Moore School of Electrical Engineering, contributed to the project reported herein:

H. E. Goheen
H. J. Gray, Jr.
C. T. Leondes

P. V. Levenian
M. Rubinoff
H. Schon


Morris Rubinoff
Project Supervisor


Harry Schon
Section Head

Approved:

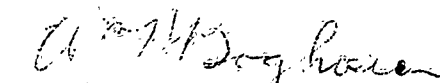

William H. Boghosian
Supervisor of Research

TABLE OF CONTENTS

PAGE

1. INTRODUCTION	
1.1 Review of Study 24-F-1	1-1
1.2 Initial Goal of the Present Study	1-4
2. INTEGRATION METHODS	
2.1 Initial Scope of the Investigation	2-1
2.2 Integration Method A	2-4
2.3 Test Roll by Method A	2-6
2.4 Ground Equations	2-6
3. IMPROVEMENTS USING ONE ARITHMETIC UNIT	
3.1 Introductory Remarks	3-1
3.2 n -Address Codes, $n < 4$	3-2
3.3 Partitioning of Memory	3-4
3.4 Auxiliary Vacuum Tube Storage Registers	3-7
3.5 Conclusions to Date	3-7
4. USE OF TWO ARITHMETIC UNITS	
4.1 Methods of Using Two Arithmetic Units	4-1
4.2 Estimated Increase in Speed	4-2
4.3 Estimated Increase in Cost and Size	4-6
4.4 Digital Differential Analyzer Study	4-6
5. HAND COMPUTATIONS	
5.1 Manoeuvres	5-1
5.2 Miscellaneous	5-1
6. CONCLUSIONS TO DATE	6-1
7. FUTURE PROGRAM	7-1

1. INTRODUCTION

1.1 Review of Study 24-F-1

A digital computer flight trainer study, numbered 24-F-1, was carried through by members of the Moore School Research Division during 1950 and 1951. A description of this study was published as Research Division Report 51-28, dated 21 March, 1951. The goal of the study was to determine the feasibility of actuating flight simulators by digital computers, it being optimistically suspected that a single digital computer might be capable of serving a number of cockpits.

As a result of study 24-F-1, the following conclusions were reached:

- (a) No existing digital computer can solve the dynamic equations of airplane flight for more than one airplane simultaneously in real time.
- (b) No existing digital computer can solve the flight equations for one airplane in real time using methods of integration in common use. In addition, a memory capacity larger than in existing machines would be necessary. However, the discrepancy is not overly large in the case of one airplane and an improved digital computer and/or improved integration methods might bridge the gap.
- (c) Digital computers show a marked superiority over analogue computers with regard to flexibility. The alteration of a flight simulator to represent a new airplane type is accomplished simply by changing an input tape and connecting to the new cockpit. Consequently the actuation of flight trainers by digital computers warrants further study.

The digital computer envisaged at the termination of the previous study is illustrated in Fig. 1.1-1. The following is a description of the figure.

- (a) Tape Library. Aerodynamics data, power plant data, and digital computer instructions for a specific airplane are placed on magnetic or punched paper tape. This tape is stored in the tape library together with tapes prepared for other airplanes. When it is desired to simulate an airplane whose tape is stored in the library, the tape is removed and placed on the tape reader. When it is desired to simulate an airplane which has just been developed, the simple, short, and relatively inexpensive operation of preparing a tape for same is performed instead of engaging in the time-consuming, involved, and expensive task of building a flight trainer for that particular airplane.
- (b) Tape Reader. The tape reader reads the information on the tape for a specific airplane and transmits that information to the digital computer.

(c) Switch "S". The reading of the tape occurs only for a short time at the beginning of the simulated flight and switch "S" then disconnects the tape reader from the digital computer.

(d) Digital Computer. The digital computer remembers:

1. Instructions, aerodynamic, and power plant data for the specific airplane.
2. Transient computational data.

The digital computer computes:

1. Aerodynamic forces.
2. Solutions of the equations of airplane motion.
3. Instrument readings.
4. Stick forces (stick forces may be computed by digital or analogue technique or a combination of both).

(e) Digital to Analogue Converters. These convert the numerical information coming from the digital computer to voltages which are applied to the instruments in the cockpit and on the instructor's control console to cause their deflections.

(f) Analogue to Digital Converters. These convert the stick and other control positions into numbers for use in the digital computer.

(g) Cockpit Selector Switch. This switch selects the cockpit which is to be activated by the digital computer.

(h) Cockpit. There is a mock-up of the cockpit for each type airplane to be simulated. When a newly developed airplane is to be simulated, a mock-up of its cockpit must be constructed and connected to the cockpit selector switch.

(i) Automatic Plotting Table. This table, controlled by signals from the digital computer, plots the simulated airplane's course.

(j) Instructor's Control Position. The instructor has before him a facsimile of the cockpit instrument readings, and controls by means of which he can simulate rough air, aircraft damage, icing, etc.

In the description above, the cockpit selector switch (g) was assumed to be manually preset at the beginning of a "flight", since the computer was assumed capable of only single cockpit operation. Should multiple cockpit operation prove to be feasible, the switch would have to be controlled directly by the computing machine itself. This direct control can be mechanized using existing techniques.

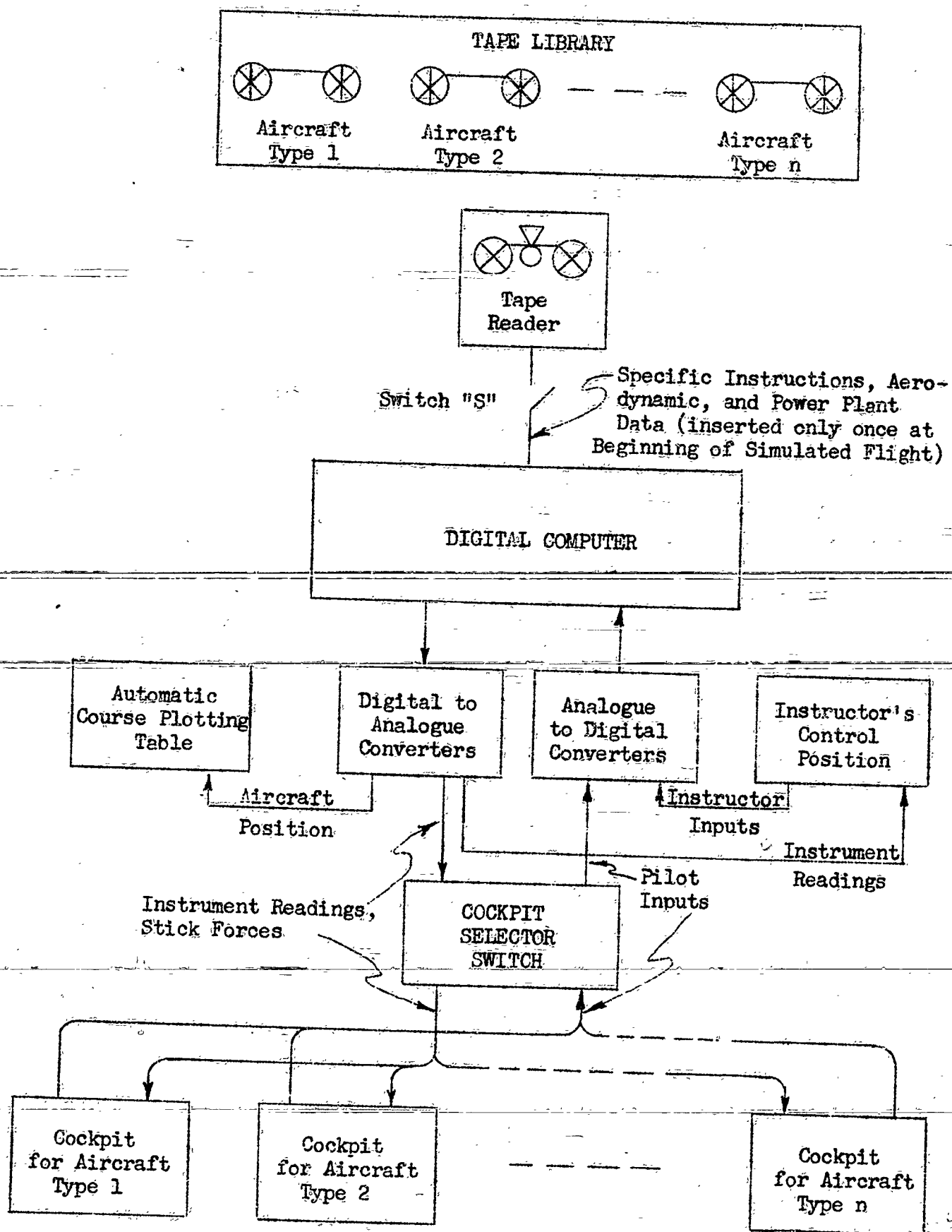


Figure 1.1-1

1.2 Initial Goal of the Present Study

The purpose of the present study 24-F-2 is to investigate methods for decreasing the computation time required per cycle and/or increasing the time interval allowable for stable integration. The goal is to be achieved assuming only existing digital computer circuit techniques, it being understood that improvements in computing machine speeds automatically render a digital simulator more feasible using even the results of the previous study.

The contents of Report 51-28 of the previous study will be drawn on heavily for background material. In particular, notions of memory requirements, number of program steps, number precision, manoeuvres, and existing methods of integration described in that report will serve as a starting point for this study.

The specific items considered worthy of investigation will now be listed. Those items marked with an asterisk were considered of particular importance and were first to receive our attention. In the following sections it will be observed that substantial progress has been made on items 1, 2, and 3. Some progress has also been made on items 4a and 6a.

1. Integration Methods

*(a) The Looking Back" Method

(b) Norlund-Type Stability Study

(c) Simplification of Ground Equations

*(d) Other Methods of Integration.

2. Improvements Using One Arithmetic Unit

(a) Partition Memory

*(b) n-Address Code, $n \leq 4$

(c) Function Tables

(d) Auxiliary Vacuum Tube Storage Registers

3. Use of Two Arithmetic Units

*(a) Problems of Sequencing Two Units

*(b) Estimated Increase in Speed for Alternative Methods

(c) Estimated Increase in Cost

*(d) Digital Differential Analyzer (Maddida) Possibilities.

4. Mixed Analogue and Digital Techniques

- *(a) Use of Analogue Integrators
- (b) Analogue \longleftrightarrow Digital Conversion
- (c) Analogue Direction Cosine Computers
- (d) Estimated Size and Cost of Analogue Equipment.

5. Aerodynamics

- (a) Separation of Equations into General Airplane Equations and Equations for Particular Airplane Types
- (b) Analysis of Relative Importance of the Various Aerodynamic Coefficients.

6. Computation

Note: A large percentage of this computation should be done by IBM or by large-scale high-speed computing machines.

- *(a) About ten Manoeuvres suspected of bad mathematical behavior, computed "accurately" to 5×10^{-5}
- ~~(b) Take-off and Landing~~
- (c) Computation by New Integration Procedures.

7. Instrument Panel

- (a) Programming of Equations to Estimate Time and Equipment Required, and to Illuminate Possible Difficulties.

2. INTEGRATION METHODS

2.1 Initial Scope of the Investigation

The previous study 24-F-1 reviewed the applicability of existing methods of integration to the flight trainer equations. It concluded that the Adams-Bashforth method incorporates some of the features suited to real time simulation problems, namely, (1) only the first derivative \dot{x} need be computed, (2) the time interval used, Δt , is constant, and (3) the integration formula is simple and therefore amenable of rapid computation. The extension of the single open integration of the Adams-Bashforth method to (the identical) open followed by repeated closed integrations of the Moulton method was then advanced as most suitable because it insured accuracy and stability.

Because the Moulton method requires repeated computation of the aerodynamic coefficients for the same time instant until all closures have been concluded, a modification was attempted. The modified Moulton method incorporated only the first three steps of the Moulton procedure, namely (1) guessing an approximate value of x'_{n+1} by open integration, (2) evaluating the new derivative \dot{x}'_{n+1} , and (3) obtaining a better approximation x''_{n+1} by closed integration. Here the process for the interval (t_n, t_{n+1}) was terminated.

Clearly both \dot{x}'_{n+1} and \dot{x}''_{n+1} , the final values accepted to be perfect, are only approximations. The only justification for the procedure was that it proved possible to merge the open and closed integration into a single integration, thus speeding up the process.

Both the Moulton and modified Moulton methods were applied to the computation of two manoeuvres during study 24-F-1. The latter of these was the test roll, a manoeuvre evidently frequently applied to airplanes to test their design characteristics. Beginning from level flight at three seconds, δ_a was varied as shown in

Table 2.1-1. The modified Moulton method proved to be unstable for intervals of $\Delta t = 1/8$ and $1/16$ sec. while for $\Delta t = 1/32$ sec. it appeared to have become stable and to agree fairly closely with the Moulton solution with repeated closures.

Table 2.1-1 Stick Motion for Test Roll

t	≤ 0	0.125	0.25	0.375	≥ 0.5
δ_a	0	.2358	.4716	.2358	0

From the results obtained, Report 51-26 concluded that the integration interval must not exceed 0.03 seconds for stable solution by the modified Moulton method, so that other methods should be investigated to permit increase of the interval. Another disadvantage was reported, which arose from the use of integrating formulae which at time t_n require derivatives back as far as t_{n-k} , $k \approx 4$. Because

a different set of parameters are used on the ground from those used in flight, the computation of derivatives of the new parameters from "past history" substantially increases the take-off and landing routines, with resultant increase in integration interval. It was thought that the use of an integration formula involving the parameters and the single derivative \dot{x}_n might reduce this difficulty.

A number of new integration methods are now to be outlined. In view of the remarks of the preceding paragraph, the first method (method A) proposed was one using parameters. Using four points, the procedure involves computing

$$(a) \quad x'_{n+1} = -\frac{10}{3} x_n + 6 x_{n-1} - 2 x_{n-2} + \frac{1}{3} x_{n-3} + 4 \dot{x}_n \Delta t \quad (2.1-1)$$

$$(b) \quad \dot{x}'_{n+1} = f(t, x'_{n+1}, S_{n+1}) \quad (2.1-2)$$

$$(c) \quad x''_{n+1} = \frac{1}{25} \left[48 x_n - 36 x_{n-1} + 16 x_{n-2} - 3 x_{n-3} + 12 \dot{x}_{n+1} \Delta t \right] \quad (2.1-3)$$

or, in words, an open integration on parameter ordinates, computation of the derivatives using the resultant approximate ordinate values but the correct pilot and instructor data, and a closed integration on parameter ordinates. \dot{x}'_{n+1} and x''_{n+1} are accepted as correct and the procedure carries on to the next interval.

Method Aa is a variation on method A which utilizes the fact that if $x^{(5)} = \frac{d^5 x}{dt^5}$ is constant over the interval, the exact

value of x_{n+1} is given by

$$x_{n+1} = \frac{2 x''_{n+1} + 0.96 x'_{n+1}}{2.96} \quad (2.1-4)$$

Thus, in method Aa, equation 2.1-4 is used to obtain a "best" value for x_{n+1} .

Method Ab extends method A even beyond method Aa. It reasons that if \ddot{x} (5) is constant, the error in \dot{x} is constant at each step.

But then the error is from equation 2.1-4,

$$E = x - x' = \frac{2 (\ddot{x} - \ddot{x}')}{2.96} \quad (2.1-5)$$

where all values of E , x , x' , and \ddot{x} are for time t_{n+1} . Then, as soon as \ddot{x}'_{n+2} is computed, a better value for computation of the derivatives \dot{x}'_{n+2} would be the sum $(\dot{x}'_{n+2} + E_{n+1})$.

Method B is directly related to the theory of least square curve-fitting. Since it is certain that the computed parameters will be somewhat in error, it is possible that a least-square fit of a polynomial of degree n to more than $(n+1)$ points might average out these errors and improve the stability of the solution. For example, four points might be least-square approximated by means of a parabola. The polynomial would then be extrapolated over the succeeding interval and integrated in the usual manner of numerical computation.

Method C is based on the sampling theorem which states that any function whose Fourier transform does not include angular frequencies exceeding Ω , may be represented exactly by the infinite series

$$x = \sum_{j=-\infty}^{\infty} x(T_j) \frac{\sin[\Omega(t + T_j)]}{\Omega(t + T_j)} \quad (2.1-6)$$

where $T_j = \frac{\pi j}{\Omega}$. Then the $(k+1)$ values of $x_n, x_{n-1}, \dots, x_{n-k}$

may be obtained from equation (2.1-6) in terms of an equal number of values of $x(T_j)$, the curtailment error being readily evaluated.

Equivalently the values of $x(T_j)$ may be solved for in terms of

x_n, \dots, x_{n-k} , and substituted for in equation 2.1-6. Finally, x_{n+1}

may be evaluated from the resultant equation. Method C as here described is readily extended to several variations.

These methods certainly do not exhaust all possibilities. In particular, another variation applicable to all methods has just been suggested, namely, the successive completion of integration for each parameter before proceeding to the next parameter integration even over the same time interval. However, methods A through C are the only ones which have received attention on this project to date.

2.2 Integration Method A

This method has already been described by equations 2.1-1, 2, 3. The primary advantage in using parameter ordinates has also been mentioned in P2.1, but this advantage has proved to be one of negligible importance in view of the alterations reported below in P2.4. On the other hand, it is possible (1) to show qualitatively why method A is more closely convergent to the true solution (i.e. more accurate) than the modified Moulton method, and (2) to prove mathematically that method A is inherently stable.

With regard to convergence, the following should be noted. If the external disturbances δ are well-behaved, then the parameter derivatives are also well-behaved. In this event the open integration for x' (equation 2.1-1) is close to the correct value of x . Consequently x' is almost correct, and hence the closed value x'' is a particularly good one. On the other hand, if the δ_{n+1} -behavior

is bad because of a "discontinuous" change, there is no method of obtaining a good x'_{n+1} . However, x'_{n+1} will absorb most of the

discontinuity since the value used at time t_{n+1} for computing x'_{n+1} is the corresponding "correct" one, δ_{n+1} . Hence x' will be

"reasonably" close to the correct value, say with error $e \ll 1$.

Moreover, equation 2.1-3 shows that the resultant error in x'' will

be only $\frac{12}{25} e \Delta t < \frac{e}{15}$ for Δt as large as $\frac{1}{8}$ second. Thus, only

a small fraction of the error appears in x'' and the latter is still a good approximation to the correct value.

It follows that the closure values obtained for the parameters are always the most nearly correct of all the terms computed. But it is precisely these values (x''_i) which are used in both equations

2.1-1 and 2.1-3. Consequently, method A should converge rather well.

With regard to stability, consider the difference equation

$$x_n + p_{n-1} x_{n-1} + \dots + p_1 x_1 + p_0 x_0 = f(t) \quad (2.2-1)$$

where the p_j are constants and $p_0 \neq 0$. There is a theorem which

states that the solution of this equation may be found in the form of a particular solution of the complete equation added to the general solution of the homogeneous equation obtained by putting $f(t)$ equal to zero.* Moreover, the latter is of the form

$$x(t) = \sum_{i=1}^n a_i (r_i)^t \quad (2.2-2)$$

where a_i are arbitrary constant coefficients determined by "initial conditions" and the r_i are the roots of the equation

$$x^n + p_{n-1} x^{n-1} + \dots + p_0 = 0 \quad (2.2-3)$$

Now, equations 2.1-1 and 3 have the form of equation 2.2-1, with $f(t)$ replaced by the term in x . Clearly, then, their solution will have a component of the form of equation 2.2-2. It is a matter of straightforward algebraic manipulation to show that there is a root R_0 greater than unity for the open integration but that the largest root for the closed expression is equal to unity.

If the solution to the flight differential equations were given exactly by both the difference equations, then the true solution would satisfy the requirement of being a particular solution of the latter. Since the integration through differences does not yield the correct solution, we may assume that at each step the computed value of x consists of two terms, the correct value X plus an error term $e(t)$. Then it follows that $e(t)$, which is not zero in general, satisfies the homogeneous equation.

In the case of the open integration, then, the solution for $e(t)$ is given by

$$e_0(t) = a_0 R_0^t + a_1 r_1^t + \dots \quad (2.2-4)$$

* "Calculus of Finite Differences" Milne-Thomson, chap. 13., MacMillan, 1933.

Since R_0 exceeds unity, it follows that $e_0(t)$ will increase exponentially with time and the finite difference solution will be unstable.

For the closed integration, the error is given by

$$e_c(t) = a_0(1)^t + a_1 r_1^t + \dots \quad (2.2-5)$$

where $|r_i| < 1$ for $i = 1, 2, \dots, n-1$. Hence the error will at worst be additively cumulative. If the error at each step be kept sufficiently small, the accumulated error for a manoeuvre (about 10 seconds of flight) should certainly remain below 10%. Hence, at least in the sense of Report 51-28, equation 2.1-3 and therefore method A is stable. (Note that with multiple closure, the method becomes stable in the broad sense).

Method A has thus been shown to be far superior to the modified Moulton method, to be convergent to simulator accuracy, and to be stable. In the next paragraph, P2.3, results obtained using method A are given.

2.3 Test Roll by Method A

The test roll manoeuvre described in P2.1 was computed at 1/8 second intervals using method A. The results are shown graphically in Figures 2.3-1 through 6 for those parameters which are not constant during the one second interval computed. The predictions of P2.2 are borne out; the results are not only stable but also as convergent as the modified Moulton method using one fourth the interval.

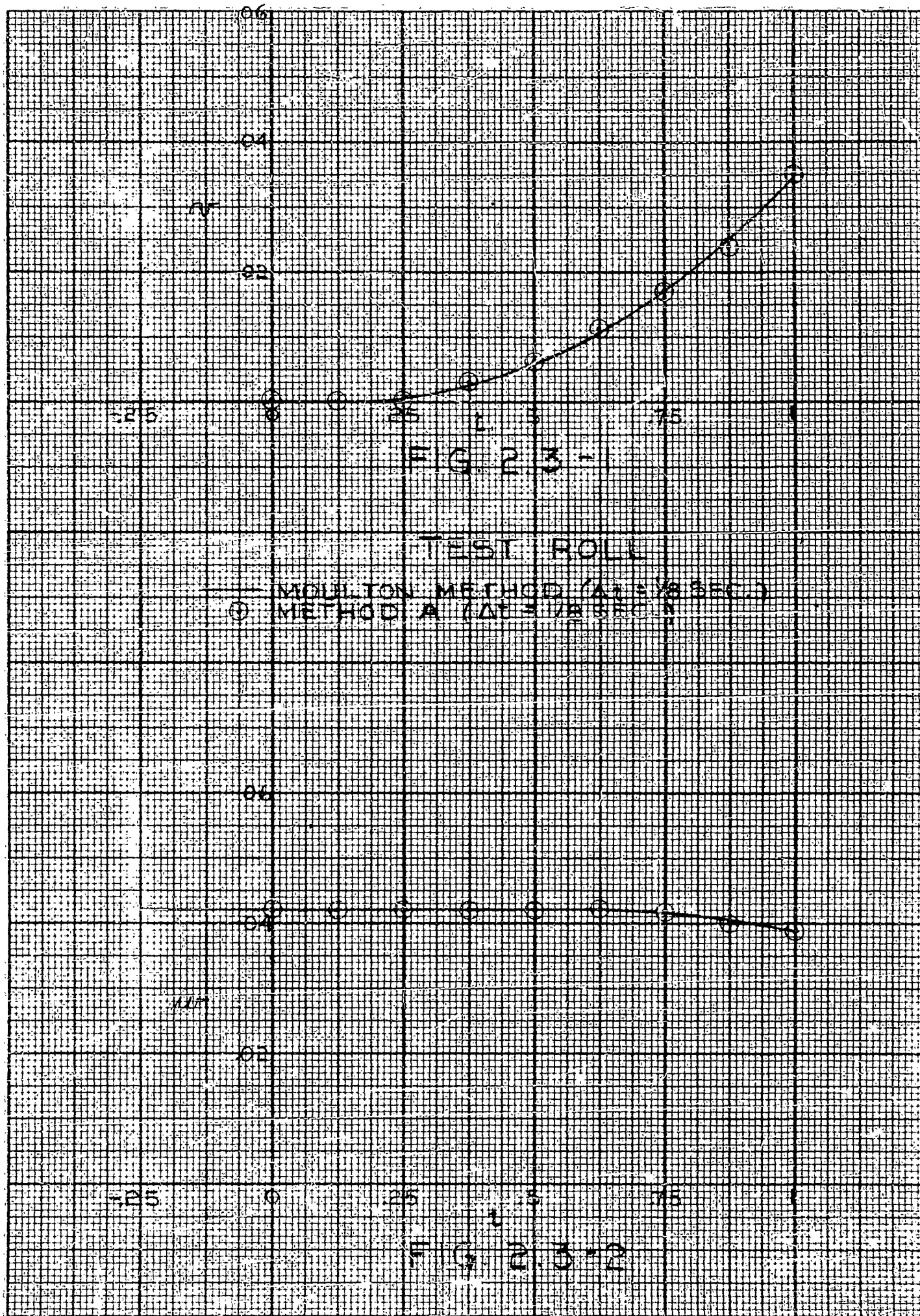
2.4 Ground Equations

In Report 51-28, subroutine 3a was inserted between subroutine 2 and 3 in order to generate past history for the flight equations at airplane takeoff. By inserting 3a into one of the ground equation subroutines, past history is computed each cycle (in general needlessly, of course) instead of in one lump as the plane takes off. The resulting reduction in the longest routine and hence in the cycle time interval is approximately 20%.

A corresponding redistribution of subroutine 10a may have a similar beneficial effect but this possibility has not as yet been investigated.

EUGENE DIETZGEN CO.
MADE IN U. S. A.

NO 340A 20 DIETZGEN GRAPH PAPER
20 X 20 PER INCH



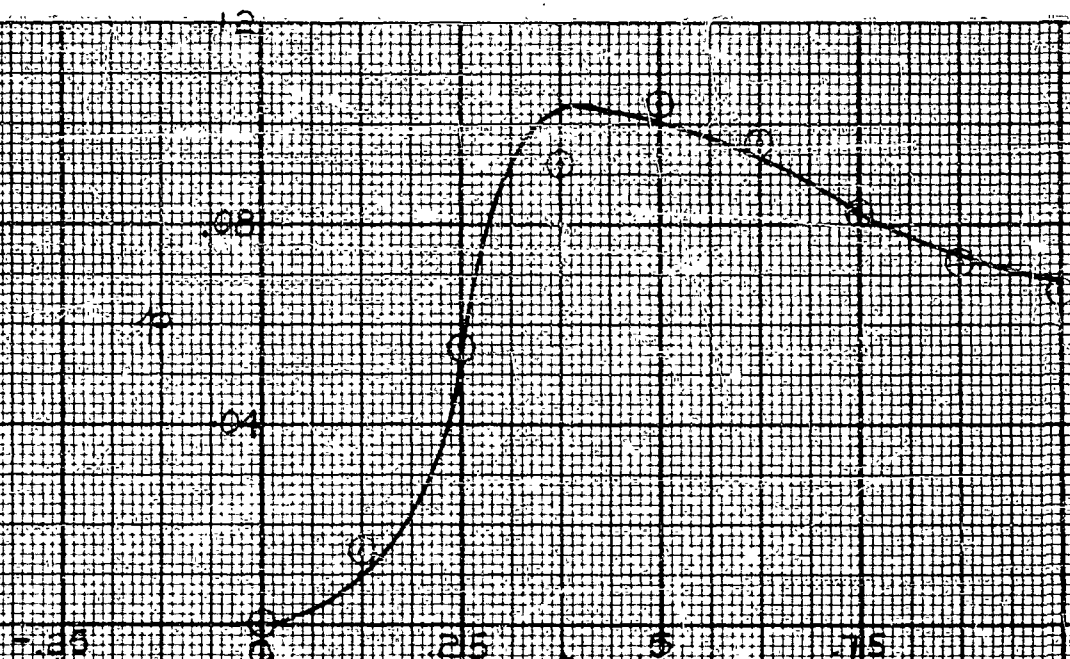


FIG. 2.3-3

TEST ROLL

— MOLLTON METHOD ($\Delta t = 1/8$ SEC.)
 O METHOD A ($\Delta t = 1/8$ SEC.)



FIG. 2.3-4

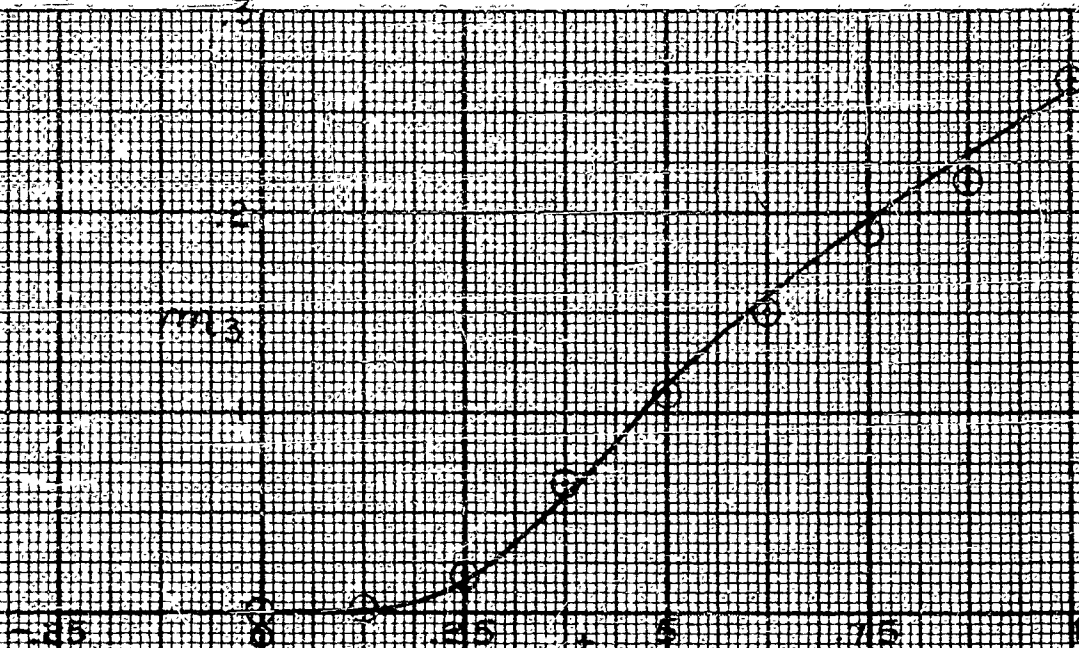


FIG 2.13-5

TEST 201

— MOUTON METHOD ($\Delta t = 1/6$ SEC.)
 ○ METHOD A ($\Delta t = 1/6$ SEC.)

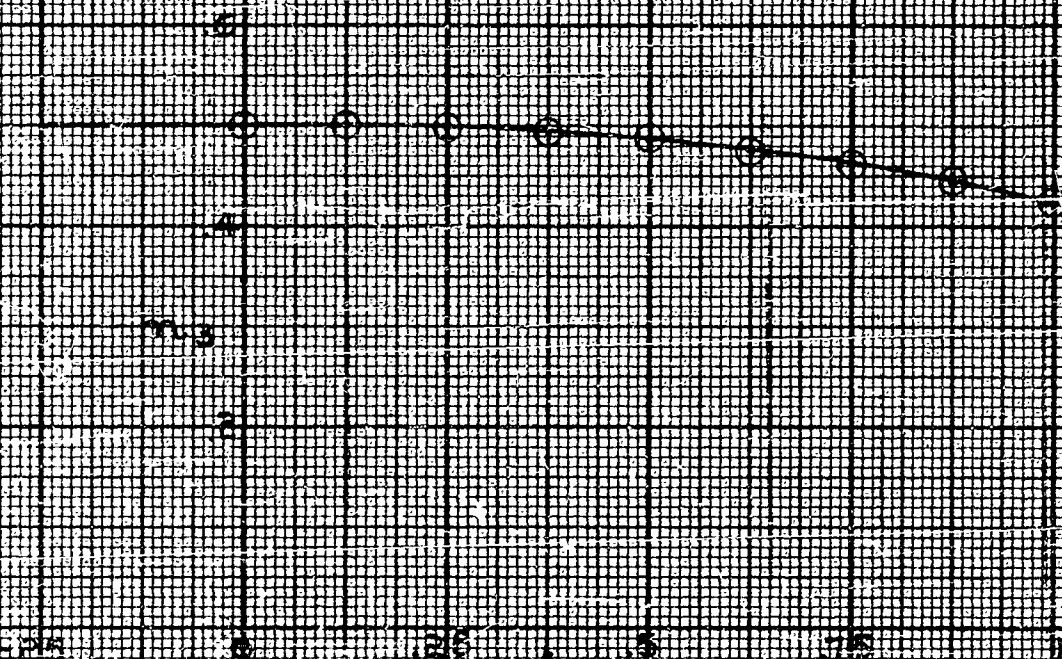


FIG 2.13-6

3. Improvements Using One Arithmetic Unit

3.1 Introductory Remarks

The previous project, 24-F-1, under contract N6onr-24913, was concluded with the finding that no existing or soon to be completed digital computer could solve the given equations of a single airplane in real time using methods of integration commonly used on digital computers.

It then becomes useful to inquire as to what machine organization would be necessary to permit solution of the given equations in real time with reasonable assurance that such machine organization is physically and economically realizable.

This section reports progress on topics having to do with improvements in machine organization using one arithmetic unit. Substantial improvement in speed has been obtained at the expense of relatively little added machine complication. The results are summarized in the conclusions in section 3.5 (see Table 3.5-1).

The questions studied are:

- (a) Partition of Memory
- (b) n-address code, $n \leq 4$
- (c) Auxiliary Vacuum Tube Storage Registers.

Serial type machines were considered over parallel type machines in what follows for the reasons below:

- (a) With the precision required for a flight trainer computer (16 digits), the speeds of the two types are about the same; if any difference exists, it favors the serial machine.
- (b) Existing serial memories are more reliable than existing parallel memories.
- (c) A parallel computer using a whipple-tree multiplier (a very high-speed serial multiplier) would be prohibitively large and costly.

In order to follow the discussion of this section it is necessary first to outline the organization of n-address machines where n may have the preassigned value 4, 3, or 1.

In the 4-address code, the order consists of 4 addresses each of which specify a unique memory location and an order type (add, subtract, etc.) Two addresses specify the location of the operands, a third address specifies the destination of the result, and the fourth address gives the location of the next order. For example, a typical instruction might be: (4, A1, A2, A3, A4); meaning "Add the contents of memory position A1 to the contents

of memory position A2; send the answer to memory position A3 and go to memory position A4 for the next instruction."

The 3-address code differs from the 4-address code only in that the location of the next order is specified by a counter which is advanced one count after the execution of an order. The counter can be selectively set by the compare orders. We would have, for example, (+, A1, A2, A3): meaning "add the contents of memory position A1 to the contents of memory position A2; send the answer to memory position A3". The location of the next order is given by the aforementioned counter. Or we could have (C, A1, A2, A3): meaning "compare the magnitudes of the contents of memory positions A1 and A2; according to which is the larger, simply advance the counter one step as usual or transfer the contents of memory position A3 into the counter". Other compare orders might of course be invented.

In the one-address code, the order consists of an address specifying a single memory location and an order type (add, subtract, etc.) The arithmetic unit in a machine using such a code accumulates the results of the computations. Time is saved because it is not necessary always to return the result of the computation at every program step to the memory. The location of the next order is specified by a counter as in the 3-address code. A typical instruction in the one-address code might be (+, A): meaning "Add the number in memory position A to the number stored in the arithmetic unit (if any) without clearing the contents of the arithmetic unit; leave the answer there and go to the memory position specified by the counter to get the next instruction".

Figure 5.2-5 of Report 51-28 (hereafter referred to simply as Fig. 5.2-5) has been used as a basis for the estimation of the longest routine time (refer to Section 5.5 in Report 51-28) using 1-address, 3-address, and 4-address codes with and without partitioning of the memory. A copy Fig. 5.2-5 has been included at the end of this section for reference purposes.

Nine different types of machines were set up (on paper). Their characteristics follow in the discussion.

3.2. n-address codes, $n \leq 4$

A 1-address machine (machine #3) was set up using the same orders that were proposed for use in the Institute for Advanced Study machine (IAS)*. Times were estimated on the basis of a word of 18 pulse times length. With a waiting period of two pulse times, this

* "Preliminary Discussion of the Logical Design of an Electronic Computing Instrument", Burks, Goldstine, von Neumann, IAS, Princeton, N. J., 1946.

allows 16 pulse times for the 16 binary digit numbers. Here, a pulse time refers to the reciprocal of the clock frequency, and hence is the time interval between corresponding points of successive clock pulses. This provides a potential 1-address capacity of 8192 memory positions and 32 order types.

3-address and 4-address codes (machines #2 and #1 respectively) were also set up on the basis of this word length. The 4-address code was the same as the 4-address code used in the counts in Section 5 of Report 51-28. The 3-address code was the same as the 4-address code except for the deletion of the 4th address and changes in the compare order to provide for conditional and unconditional transfers (Fig. 5.2-5 requires no compare orders).

Counts were made on simple sequence 3-3 using these codes. Pulse times are tabulated in Table 3.2-1.

Table 3.2-1 Pulse Counts on Machines 1 - 4 for SS3-3

Machine No.	1	2	3	4
Addresses	4A	3A	1A	1A
Pulse Times	2660	2480	2340	2020

It became apparent when the counts were made that the 1-address code used above wasted a great deal of time. A new 1-address code (machine #4) was constructed consisting of 20 orders. Pulse times for SS3-3 and this machine are also tabulated in Table 3.2-1. An improvement is evident and it was thought worthwhile to estimate the pulse times required for the entire routine on the basis of Fig. 5.2-5. The results are recorded in Table 3.2-2.

Table 3.2-2 Estimated Pulse Counts for Entire Routine

Machine No.	1	2	4
Addresses	4A	3A	1A
Pulse Times	237,000	218,000	176,000

3.3 Partitioning of Memory

While the above figures show that the 1-address code machine is fastest, certain inconsistencies in setting up the codes and estimating times could have swung the verdict in favor of one of the other codes. Also it was desired to see if partitioning the memory could be used to advantage. Further counts were then undertaken.

Since the problem of the F9F as programmed in the Flight Trainer Report took about 5000 memory positions for instructions and 496 memory positions for numerical information, excluding outputs, it was felt that a possible saving in address length (and computing time) could be achieved by partitioning the memory into a section containing only numbers and a section containing only orders. This would result in shorter addresses since most of the addresses in orders refer to the number storage places and not to the order storage places.

It was assumed that a number storage of 1024 memory positions, and an order storage of 8192 memory positions would be adequate for most flight trainers. The composition of the orders in 1A, 3A, and 4A codes would then consist of the number of bits shown in Table 3.3-1.

Table 3.3-1 Bits per Order Word with Partitioned Memory*

Code	Operand 1	Operand 2	Result	Next Order	Order Type	Total
1A	10	x	x	x	5	15
3A	10	10	10	x	4	34
4A	10	10	10	13	4	47

* Number length: 15 bits + sign = 16 bits.

In addition switching time need not be specially provided for extraction of orders from the memory since this is automatically provided by the time taken to extract numbers from the memory and to complete the arithmetic operation, but switching time is needed for extracting numbers in the 3A and 4A codes.

Switching time is estimated to be 2 pulse times. Although Hurricane allows about 100% of the word time for switching, MSAC and SEAC allow only 10%. As a result, the number of bits per order word and per number word are as displayed in Table 3.3-2.

Table 3.3-2 Total Bits per Order and per Number Word

Code	Order Word	Number Word	Switching Blank	Number Word Total
1A	15	16	0	16
3A	34	16	2	18
4A	47	16	2	18

It was then assumed that the number and order words had the same length in pulse times so that the number and order memories could maintain synchronism. The smallest number of memory positions of one number length that would hold an order is then 1, 2, and 3 for the 1A, 3A, and 4A codes, respectively. The resultant word length is shown in Table 3.3-3.

Table 3.3-3 Number and Order Word Lengths (Pulse Times)

Code	Number Word	Order Word
1A	16	16
3A	18	36
4A	18	54

The compare order in the 1A code deserves special mention. There are 8192 order memory positions, but the address can specify only 1024 of them (possibly 2048). Consequently it is possible to enter the order memory in only 1024 places when directed to do so by the compare orders (conditional and unconditional transfers). This is not expected to be any great handicap as the program for the F9F flight routine required only about 29 compare orders. Many possible ways of performing this order have been considered, all of them feasible and of varying degrees of flexibility. It was considered too soon to study the relative merits of these methods.

The time for one computation cycle was computed in the Flight Trainer Report for a routine that involved 356 multiplications and only 141 additions. It was felt that computing time could be greatly reduced by speeding up multiplications. The so-called "whipple-tree multiplier" using 14 adders can multiply

in two word times*. Operation times were determined on the basis of a whipple-tree multiplier. Pulse times counted for Fig. 5.2-5 are shown in Table 3.3-4.

Table 3.3-4 Pulse Counts for Fig. 5.2-5

Machine No.	7	6	5
Code	1A	3A	4A
Pulse Times	10,210	14,580	17,080

The entire routine took about 0.1 sec. and Fig. 5.2-5 took 14,840 microseconds when time was computed on the basis of a 16 digit parallel computer in the Flight Trainer Report. The routine pulse times can then be estimated by multiplying the pulse times for Fig. 5.2-5 by $100,000 \div 14,840 = 6.74$ with the results shown in Table 3.3-5.

Table 3.3-5 Pulse Times for Longest Routine

Machine No.	7	6	5
Code	1A	3A	4A
Pulse Times	68,700	98,200	115,100

The time in seconds per routine is obtained by dividing the above pulse times by the clock repetition frequency in cycles per second (see Table 3.5-1).

Since the UNIVAC is in reliable operation at 2.25 mc, it is felt that the speed of 0.031 seconds per routine is attainable with present-day techniques in a digital flight trainer. The verdict above is definitely in favor of a 1A code.

* There are other multipliers which can multiply in two word times. The choice of the best multiplier is left for further consideration at a later date, with use in speeding division a particularly influential parameter.

3.4 Auxiliary Vacuum Tube Storage Registers

During the progress of the work above, it was observed that in the computation of a simple sequence (the numbered, blocked-in computations on the program diagrams) there was no reason for returning any computed quantity to the memory except at the end of the simple sequence. One advantage of the LA code is that it reduces the need for returning information to the memory, but the reduction is not complete. It was felt that inclusion of extra registers in the arithmetic unit would be of value.

Since an order always carries an address which specifies a memory location, it would be desirable to extract an order from the order memory only when it is desired to insert or remove quantities from the number memory. However, for example, a number may be removed from the memory, multiply a number stored in the arithmetic unit, and the product be added to another number stored in an auxiliary register, the result to remain in the arithmetic unit. It would be desirable to accomplish all this with one order - a "composite" order. Such a composite multiply-add order was set up (machine #8) and the routine times were estimated in the above manner. (See Table 3.5-1). A small improvement is evident.

There is apparently a large number of composite orders of this type that can be set up. If the most all-inclusive "composite" order be utilized, the routine times become those of machine #9 in Table 3.5-1.

3.5 Conclusions to Date

Table 3.5-1 summarizes routine times for the different machine organizations. These figures, however, should not be taken as final but rather as approximated estimates.

A direct comparison of routine times with and without memory partition is not shown. However, that there is an improvement in speed at negligible increase in complication is felt to be the obvious result of partitioning the memory in the manner described.

At this time, a machine to be incorporated in a flight trainer would have the following characteristics (machine #8):

- (a) Serial binary.
- (b) Acoustic delay line memory.
- (c) 1-address code.
- (d) 16-digit word.
- (e) Clock frequency of about 2.25 mc.
- (f) Separate order and number memories.
- (g) Whipple-tree or other multiplier having high speed.
- (h) Use of composite order: Multiply-add. It is felt at this time that the gain in speed by full use of composite orders is outweighed by the added complication (and cost) resulting therefrom.

Table 3.5-1 Routine Times for All Machines

Machine No.	Partition Memory?	Whipple-tree Multiplier?	Composite Orders?	Code	Routine Time-Seconds**			Comments
					1 mc	2.25 mc	4 mc	
1	No	No	No	4A	.237	.106	.059	As in Flight-Trainer Report
2	No	No	No	3A	.218	.097	.054	As in machine #1 with 4th address deleted
3	No	No	No	1A	.176*	.078*	.043*	IAS Code (code only)
4	No	No	No	1A	.176	.078	.043	Modified Code
5	Yes	Yes	No	4A	.115	.051	.029	See section on memory partition
6	Yes	Yes	No	3A	.098	.044	.025	"
7	Yes	Yes	No	1A	.069	.031	.017	"
8	Yes	Yes	mult.-add	1A	.062	.028	.016	See section on additional registers
9	Yes	Yes	full use	1A	.056	.025	.014	"

* Minimum possible value

** The 1 m.c. rate refers to a computer similar to the SEAC or to the Moore School MSAC; the 2.25 and 4 m.c. rates to the UNIVAC and HURRICANE, respectively.

4. USE OF TWO ARITHMETIC UNITS

This section is devoted to a discussion of the possible improvement in digital computer speed as a result of using more than one arithmetic unit. In order to facilitate discussion it is useful to define the following terms:

- (a) Program step: One basic machine operation (add, subtract, multiply, divide, compare, shift, or etc.)
- (b) Simple sequence: One or more program steps used in the computation of a single equation.
- (c) Compound sequence: A group of program steps used in the computation of more than one equation.
- (d) Branch point: A point where the succeeding program step is selected from one of two alternative program steps, or where the succeeding program step is entered from one of two alternative program steps.
- (e) Flight branch point: A branch point where a selection is made between two alternative modes of flight or where entry is made from two alternative modes of flight.
- (f) Sub-routine: A compound sequence embracing all simple sequences between two flight branch points.
- (g) Routine: A basic sequence of sub-routines the sequential repetition of which describes a motion of the airplane.

4.1 Methods of Using Two Arithmetic Units

There presumably are a large number of methods of programming for a computer using two arithmetic units. Five methods of greatly varying merit will now be outlined.

In the first method the two units share the computation of each simple sequence. When the work on the first simple sequence of a subroutine has been performed, the two units proceed to the next, and so on, until all of the simple sequences within the sub-routine have been computed. When a flight travel point is reached, the two units start the computation of the first simple sequence of the next sub-routine. This method involves a great deal of idling time (time when one unit is not performing any program step) within the simple sequences, since in the course of computation one unit must in many cases wait for the result of the other.

In the second method, one unit works on one simple sequence while the other unit works simultaneously on another simple sequence within the same sub-routine. This method involves no idling time within the simple sequences. However, one simple sequence may depend

on the solution of another. Hence, one unit may have to wait for the result of the other before proceeding. Since there are many independent simple sequences within a sub-routine, this idling time can be held to a value much lower than that of method one.

The third method is a combination of the first two. Thus, in method one, instead of one unit waiting for the result of the other, it proceeds immediately to some part of another simple sequence. By programming the two units to jump back and forth among the many program steps within a sub-routine, the idling period can be held to a minimum. It can be seen that this method is more versatile than the first two.

In a fourth method, one unit computes one sub-routine while the other unit works simultaneously on another sub-routine. In order to use this method the sub-routines to be computed must be known beforehand. In many cases, however, the decision as to which will be the succeeding sub-routine is made at the end of the sub-routine being computed. Hence, often, the sub-routines to be used can not be determined beforehand. In these cases one unit must wait for the other to compute an entire sub-routine before it can proceed. In the cases where two sub-routines can be computed simultaneously the problem of idling is still prominent, since in general the solution of one sub-routine depends on results obtained from the simple sequences of previous sub-routines.

The fifth method is an extension of method three to include more than one sub-routine. Where it is possible for two sub-routines to be computed simultaneously, the two units perform the many program steps included in both sub-routines. Where the sub-routines used are based on decisions made at the end of previous sub-routines, the two units compute within a single sub-routine, i.e., method three is inherently included in this method. Because of its versatility this method has the least amount of idling time.

The criterion for a choice among the above methods is the time that it takes for the two units to compute the routine. The best method is the one through which the smallest length of time is obtained for the routine computation. According to this criterion method five is the best to use since it contains the smallest amount of idling time. Method two is also promising and can probably be used to advantage. Methods one and four can be eliminated because of their large idling times, while the third method is included in method five.

4.2 Estimated Increase in Speed

In discussing speed it is useful to introduce a figure of merit, F , defined by the following relationship:

$$F = \frac{T_1}{T_N} \quad (4.2-1)$$

where T_1 is the time it takes to compute a routine using one unit

and T_N is the time it takes to compute the same routine using N

units. The minimum value of the figure of merit for any number of units (N) is one, since it can never take more time to compute a routine using N units than it does using one unit. The maximum value of the figure of merit for a given N is equal to N . For example, using two units ($N = 2$) and assuming perfect programming (zero idling time), the computation of the routine can be evenly divided between the two units and the time for each unit to perform its work is exactly one half the time it takes one unit to compute the routine. Since the two units work simultaneously, the figure of merit is equal to two. In the general case where the idling time is not equal to zero, the figure of merit lies between one and N .

Parts of sub-routine number one (see Report 51-28) were programmed using method five. It became apparent that the necessity for the two units to share the memory input and output busses was the main cause of idling time. Thus, while instructions or numbers are being withdrawn from the memory for one unit it is not possible at the same time to withdraw instructions or numbers for the other unit. One solution to this problem was to have two memory input-output systems. However, this method was discarded because of the apparently disproportionate increase in the size and cost of the digital computer. Having decided that the memory busses were to be shared, the following recourse was the only one left open: while one unit is performing an order, the instruction and numbers for the other unit are given access to the memory bus.

It is useful at this point to define two more terms. First, access time is defined as the time when information appears on the memory input or output busses. Second, computation time is defined as that time during which a unit is performing an instruction when no information for its use appears on the memory input or output busses.

The effect of bus sharing on the figure of merit will now be shown for two units. Of the time that it takes one unit to compute the routine, let A be the access time, and let C be the computation time. It follows that

$$T_1 = A + C \quad (4.2-2)$$

For the length of time, T , that it takes two units to perform the computation of the routine, there are two distinct cases: (a) $A \geq C$ and (b) $C > A$.

For the case where $A \geq C$, T_2 is equal to A . This can be seen from Figure 4.2-1a. The access times of the two units are added to give the left-hand column. Since the access time of the two units cannot occur simultaneously, their resultant addition is (no less than) A . The computation times and access times are programmed to occur simultaneously, as shown in the figure. Since A is greater than C there must

be some time in the right-hand column when nothing is done. This is the idling time. For $A \neq C$, then, the figure of merit is

$$F = \frac{T_1}{T_2} = \frac{A + C}{A} = 1 + \frac{C}{A} \quad (4.2-1a)$$

and the effect of bus sharing is to limit the figure of merit to a value less than two.

A	C
	Idling Time

(a)

A	$\frac{A}{C} \times C$
	$\frac{C-A}{C} \times C$

(b)

A	$\frac{A}{C} \times C$
$\frac{C-A}{2C} \times C$	$\frac{C-A}{2C} \times C$

(c)

Figure 4.2-1

For the case where $C > A$, T_2 is equal to $\frac{A + C}{2}$. This can be seen from Figure 4.2-1b and c. The total access time A and $\frac{A}{C}$ of the computation time occur simultaneously. The remainder of the computation time $\frac{C - A}{C} \times C$ can be divided equally between the units since computation times can occur simultaneously. Hence

$$T_2 = A + \frac{C - A}{2} = \frac{A + C}{2} \quad (4.2-3)$$

Thus, for $C > A$, the figure of merit is

$$F = \frac{T_1}{T_2} = \frac{A + C}{\frac{A + C}{2}} = 2 \quad (4.2-1b)$$

When $C > A$ bus sharing does not limit the figure of merit. Another conclusion which can be drawn is that for $C > A$ more than two units may be used to advantage.

It is well to point out that the figures of merit found by the above equations may not actually be obtainable due to programming difficulties. However, the figures of merit found in practice can never be greater.

A and C for the flight trainer problem can be found in the following manner. First, separate totals for all the different types of orders used in the routine under consideration must be found. These totals for the flight trainer problem can be found in Figure 5.3-1 of Report 51-28. The second step is dependent on the particular computer under consideration. For a specified computer, the time it takes the computer to perform each instruction must be found. Then for each instruction the percentage of this time which is access time and the percentage which is computation time must be found. These times will evidently vary from one computer to the next, depending on their logical characteristics. The first and second steps are next combined to find A and C for the given routine using a specific computer. Knowing A and C the limitation of the figure of merit due to bus-sharing can be determined.

Using the above method, limiting figures of merit were found for three different types of computers using two arithmetic units and operating on the routine designated by the sequence of sub-routines 1-2-3a-3-9. The results are tabulated in Table 4.2-1.

Table 4.2-1 Figures of Merit for Two Arithmetic Units

Computer	$\frac{A}{A+C}, \%$	$\frac{C}{C+A}, \%$	Figure of Merit
Hurricane	45.8	54.2	2
I.A.S.	47.0	53.0	2
Machine #7 of Section 3.3	74.1	25.9	1.35

For the Hurricane and IAS machines, the fact that C is greater than A indicates that more than two units might possibly be used to advantage. If only two units are used, it indicates that the figures of merit are not limited by bus-sharing. For machine #7 of Section 3.3 it is seen that the figure of merit is severely limited by bus-sharing.

4.3 Estimated Increase in Cost and Size

Clearly, it is important to estimate the increase in cost and size of the machine due to the second arithmetic unit. Naturally this increase varies with machines which differ in circuit and/or logical design. The estimate made here is for a machine similar to the Moore School's MSAC.

The method used was to determine which units of the machine would be duplicated in order to accommodate two arithmetic units. The units were the following: the dispatcher memory loop, the execute selector counter, and the order type selector. With this assumption and assuming certain increases in the size of the timer and of the power supply, it was estimated that a one-third increase in both cost and size would result.

4.4 Digital Differential Analyzer Study

A theoretical study of the MADDIDA computer was undertaken which had as its objectives the determination of (1) using a separate MADDIDA as a second arithmetic unit just to perform the integrations, and (2) employing some aspects of digital differential analyzer techniques in the solution of this problem.

We shall now outline briefly the basic characteristics of the only existing digital differential analyzer, the MADDIDA (Magnetic Drum Digital Differential Analyzer). This is a serial magnetic drum

instrument. Although the memory unit can be speeded by employing a mercury delay line, it will be seen that the serial nature of the machine is inherent in MADDIDA.

Any number A is represented in MADDIDA by the mapping $(1 + A)$ with the restriction $-1 \leq A < 1$. Numbers are represented to a possible precision of 29 binary places.

The method of integration employed in MADDIDA is that of direct summation of the integrand at uniformly spaced values of the argument, i.e.

$$z = k \sum_i (y_i \, d x) \quad (4.4-1)$$

where k is a constant associated with the integrator, z is the output of the integrator, y_i is the integrand, and x is the independent variable. Since y is the accumulation, or sum, of $d y$'s, this equation can be expressed as

$$z = k \sum_i (d x \sum d y)_i \quad (4.4-2)$$

Clearly, the integration is a linear one. Consequently, the curtailment error per step is of the order of magnitude $(\Delta x)^2 f''(\xi)$. From this it follows that although MADDIDA is capable of a precision of 29 binary places, its accuracy is limited by curtailment to somewhat less than 29 binary places, unless a particularly small interval Δx be used.

The maximum rate of $d z$ output of an integrator is 1 pulse per cycle of computation. The rate of y increase of an integrator is restricted in MADDIDA to a maximum of three (lowest order) binary columns per cycle of computation, the actual increase depending on the number of integrators supplying the integrator under consideration.

The time taken per cycle of computation is determined by the speed of a cycle of the magnetic drum memory. The magnetic drum in MADDIDA rotates at about 100 r.p.s., so that a cycle lasts 0.01 seconds. In part 3.7 of Report 51-28, it is mentioned that a precision of 15 binary places is required. Since the (Δx) required to keep the curtailment error within these limits is apparently so small as to preclude the possibility of a high rate of increase for many of the variables involved, it is clear that the rate of integration is much too slow for flight trainer use.

An integrator of MADDIDA may be used to perform the operation of addition. However, the method is essentially equivalent to addition.

by counting which is inherently much slower than by addition in an accumulator. Hence there is no point in discussing the MADDIDA adder here.

Multiplication is carried out by "connecting" integrators exactly as in continuous variable differential analyzers. For instance, to obtain (uv) one would solve $\left[\int v du + \int u dv \right]$.

Besides performing the operations of integration, addition, and multiplication, a MADDIDA integrator may be coded as a servo. This is useful in inverting operations and it is sometimes necessary to achieve proper scaling.

For the insertion of empirical data into the computation, 12 empirical input channels are provided in the machine. These input channels may be fed from various input devices.

The control specifying which problem is being solved is achieved by employing 2 channels, L_1 and L_2 , which specify (a) the interconnection of the integrators and (b) which input channels are being employed. An additional "z"-channel contains the results of the preceding cycle of computation and feeds this information to the various integrators as directed by L_1 and L_2 .

At the moment it is felt that MADDIDA techniques are too slow to be employed in a simulator. The major reasons are (1) the slow pulse repetition rate because of the memory employed, (2) the use of time-sharing circuits, (3) the extremely short time interval needed to obtain reasonable accuracy in the method of integration employed.

As regards reason (1), it has already been mentioned that the memory could be mercury delay line. Since there are 10 independent differential equations to be solved, a cycle of computation would include 10 integrators. For a 4 m.c. pulse rate and 16 digits per word, this implies a time of the order of magnitude of 40 microseconds per cycle. This speed is certainly adequate if Δx may be chosen large enough.

As regards reason (2), it is not necessary to operate all 10 integrators serially in this way. With parallel operation, the time per cycle may be reduced, but only at added expense.

As regards reason (3), the interval Δx required to keep the curtailment errors within reasonable limits is too small. This implies that the computation of the aerodynamic coefficients must also be undertaken for each time instant for insertion as the increase in the integrand of an integrator. Thus, the time might be reduced at most to that needed to compute these coefficients. This latter time is at present too large for the Δx required.

The final conclusion therefore is that MADDIDA with the best of existing computer techniques is inapplicable for direct use in the flight trainer problem. Possible use of some MADDIDA features will receive further consideration. However, it is felt that even these efforts will not prove to be fruitful. Therefore, decreased effort will be expended in this direction.

5. HAND COMPUTATIONS

5.1 Manoeuvres

The purpose of the hand computation of manoeuvres is to provide starting data for eventual computation on a high-speed automatic computing machine. At the same time these initial results may be scanned for suspected instability owing to improper stick and rudder information being assumed.

The manoeuvres which have been begun are the high speed roll and the outside loop. Level flight at 25000 feet has been modified by decreasing the assumed airplane weight, W , below unity. Previously the value 1.07 had been used.

5.2 Miscellaneous

Computation of the test roll by method A has been computed. The results have been reported in section 2. The test roll is being computed by the Moulton method using fewer differences in order to determine the influence of the approximation polynomial employed in integration on the flight parameters.

Computations have also been undertaken in connection with an initial study of the aerodynamic coefficients. It is thought that time may be saved by simplifying computation of experimental aerodynamic coefficients by use of, for example, a multi-dimensional trigonometric expansion of few terms. A function table containing $\sin x$ would then be an aid to computation. The study has started with the drag coefficient for the F9F. To date nothing has been found that is superior to the straight line approximation method described in the 24-F-1 report.

6. CONCLUSIONS TO DATE

As a result of the investigations completed thus far, the following three conclusions appear to be justified:

- (a) A computation interval $\Delta t = 1/8$ second appears feasible as a result of integration method A. This conclusion rests both on mathematical theory and on the results of one numerical computation of a "discontinuous" airplane motion. (See section 2).
- (b) The time ΔT to compute the longest routine can be decreased to less than the value of 0.1 seconds considered minimum by Report 51-28. The improvements possible are illustrated in Table 3.5-1 for machine 7, 8, and 9 of the MSAC, UNIVAC, and HURRICANE types, all using 15 bit number precision. It is important to observe that all these machines use only existing circuit design, gaining their improved speed from novel logical structure and programming.
- (c) On the strength of the preceding conclusions, it appears definitely feasible to actuate at least one flight trainer cockpit by a digital computer.

7. FUTURE PROGRAM

Further investigations are to be pursued along the lines of the initial program described in section 1. The other methods of integration will be looked into, particularly the variations on method A. A decelerated study on improvements using one and two arithmetic units will be carried out. It is planned to postpone intensive investigations of aerodynamic coefficients and instrument panel data to a later point in the project, except for some preparatory work on the former.

Emphasis will be placed upon the following studies:

- (a) Other airplane types; the effect of airplane type on the behavior of aerodynamic coefficients, on the interval Δt required to compute the longest routine, on the character of the program for ground motion and for flight, and on the size of both the number and the instruction memories required.
- (b) Manoeuvres; the preparation of programs for computing manoeuvres on automatic machines, including the specification of motions of stick, rudder, throttle, etc.
- (c) Error analysis on the methods of integration; either theoretical or empirical or both.
- (d) Aerodynamic coefficients; the computation of the effect of altering an aerodynamic coefficient by say 10% on a given manoeuvre or manoeuvres.

REEL-7402

A.T.I. 209023

Reproduced by
DOCUMENT SERVICE CENTER
ARMED SERVICES TECHNICAL INFORMATION AGENCY
KNOTT BUILDING, DAYTON, 2, OHIO

"NOTICE: When Government or other drawings, specifications or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the U.S. Government thereby incurs no responsibility, nor any obligation whatsoever; and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications or other data is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use or sell any patented invention that may in any way be related thereto."

UNCLASSIFIED

23/5
30/2

STI-ATI-209 023

237000

UNCLASSIFIED

p 1/4

Moore School of Electrical Engineering, Pennsylvania U.,
Philadelphia.

FEASIBILITY OF ACTUATING TRAINERS BY DIGITAL COMPUTERS, by Morris Rubinoff and Harry Schon. Progress rept. no. 1 for 1 Feb 52. 1v. incl. illus. tables. (Research rept. no. 52-22)(Contract N6onr-24915)

SUBJECT HEADINGS
Computers, Digital
Trainers, Flight

DIV: Personnel & Training (23)
SECT: Military Training & Indoctrination (5)
DIV: Research & Research Equipment (30)

SECT: Computers (2)

DIST: Copies obtainable in ASTIA-BSC

UNCLASSIFIED

CFST# per ONR ltr,
27 Apr 66

see →

8. FLIGHT TRAINERS

X FLIGHT CONTROL COMPUTERS

23/5

STI-ATI-209 023

UNCLASSIFIED

30/2

Moore School of Electrical Engineering, Pennsylvania U.,
Philadelphia.

FEASIBILITY OF ACTUATING TRAINERS BY DIGITAL COMPUTERS, by Morris Rubinoff and Harry Schon. Progress rept. no. 1 for 1 Feb 52. 1v. incl. illus. tables. (Research rept. no. 52-22)(Contract N6onr-24915)

SUBJECT HEADINGS

DIV: Personnel &
Training (23)

Computers, Digital
Trainers, Flight

SECT: Military Training &
Indoctrination (5)

DIV: Research & Research
Equipment (30)

SECT: Computers (2)

DIST: Copies obtainab  m ASTIA-DSC

UNCLASSIFIED